WHAT IS CLAIMED IS:

1. A processor having a normal mode and a speculative prefetching mode, the processor operable in the speculative prefetching mode after a data cache miss, comprising:

a first data cache for storing data when the processor operates in the normal mode; and

a second data cache for storing data in response to a store instruction when the processor operates in the speculative prefetching mode.

2. The processor of claim 1, wherein the second data cache comprises:

an entry for storing data; and

a trash bit associated with the entry, wherein the trash bit indicates whether the entry contains arbitrary data.

3. The processor of claim 2, further comprising control logic for setting the trash bit of the second data cache when source operand for the store instruction depends on the data cache miss.

4. The processor of claim 3, wherein the second data cache further comprises a valid bit associated with the entry, the valid bit for indicating whether the entry is in use during the speculative prefetching mode.

5. The processor of claim 4, further comprising control logic operable for reading data from the entry of the second data cache in response to a load instruction provided the valid bit is set, the control logic operable for reading data from the first data cache in response to the load instruction provided the valid bit is clear.

6. The processor of claim 1, further comprising:

a first register for storing data when the processor operates in the normal mode;

a second register for storing data in response to a load instruction when the processor operates in the speculative prefetching mode, the second register further having a trash bit for indicating whether the second register contains arbitrary data; and

control logic for setting the trash bit of the second register when source operand for the load instruction depends on the data cache miss.

7. The processor of claim 6, wherein the first register and the second register are located in separate register files of the processor.

8. The processor of claim 6, wherein the second register comprises a valid bit for indicating whether the second register contains data.

9. The processor of claim 8, further comprising control logic operable for reading data from the second register in response to an instruction provided the valid bit is set, the control logic operable for reading data from the first register in response to the instruction provided the valid bit is clear.

10. The processor of claim 1, wherein the first data cache comprises a direct-mapped cache and where the second data cache comprises an associative cache.

11. A processor having a normal mode and a speculative prefetching mode, the processor operable in the speculative prefetching mode after a data cache miss, comprising:

a first register for storing data during the normal mode;

a second register for storing data during the speculative prefetching mode, the second register comprising a first trash bit that indicates whether the second register contains arbitrary data;

an instruction bus for receiving a stream of instructions including a first instruction and a second instruction;

control logic for executing the first instruction;

control logic for initiating a cache fill request provided execution of the first instruction encounters a data cache miss;

control logic for setting the trash bit of the second register in response to the first instruction and the data cache miss;

control logic for executing the second instruction in the speculative prefetching mode using the second register in place of the first register.

12. The processor of claim 11, further comprising:

a first data cache for storing data during the normal model; and

a second data cache for storing data in response to a store instruction during the speculative prefetching mode.

13. The processor of claim 12, wherein the second data cache comprises:

an entry for storing data; and

a second trash bit associated with the entry, the second trash bit for indicating whether the entry contains arbitrary data.

14. The processor of claim 13, further comprising control logic for setting the second trash bit in the second data cache when source operand for the store instruction depends on the data cache miss.

15. The processor of claim 14, wherein the second data cache further comprises a valid bit associated with the entry, the valid bits for indicating that the entry is in use during the speculative prefetching mode.

16. The processor of claim 15, further comprising control logic for reading data from the entry of the second data cache in response to a load instruction provided the valid bit is set, and for reading data from the first data cache in response to the load instruction provided the valid bit is clear.

17. The processor of claim 11, further comprising control logic operable for updating a branch prediction state for a branch instruction in the speculative

prefetching mode provided the branch instruction is not dependent on arbitrary data.

18. The processor of claim 17, wherein the branch prediction state is not updated for the branch instruction during the speculative prefetching mode provided the branch instruction is dependent on arbitrary data.

19. The processor of claim 11, wherein the first data cache is a direct-mapped cache and the second data cache is an associative cache.

20. In a processor having a first cache memory for storing data in a normal mode and a second cache memory for storing data in a speculative prefetching mode, a method of performing memory accesses, comprising:

(a) receiving a first load instruction that calls for transferring a first data into a first register;

(b) determining whether the first data is present in the first data cache;

(c) provided the first data is missing from the first data cache, initiating a first cache fill request to retrieve the first data from an external memory, copying a first program counter to a second program counter, and writing an arbitrary data into a special register such that program execution continues with the second program counter and the arbitrary data;

(d) before the first cache fill request is completed, receiving a second load instruction that calls for transferring a second data into a second register, and determining whether the second data is present in the first data cache and the second data cache;

(e) provided the second data is missing from the first data cache and the second data cache, initiating a second cache fill request to retrieve the second data from an external memory;

(f) after the first cache fill request is completed, resuming program execution using the first program counter and ignoring the arbitrary data stored in the second register and in the second cache memory; and

(g) re-executing the second load instruction, wherein retrieval time of the second data is reduced as the second cache fill request has been previously initiated.

21. The method of claim 20, further comprising:

before the first cache fill request is completed, receiving a store instruction that calls for transferring the first data from the first register to an external memory; and

in response to the store instruction, transferring the arbitrary data from the special register to the second cache memory.